Fitness Clouds and Problem Hardness in Genetic Programming

Leonardo Vanneschi, Manuel Clergue, Philippe Collard, Marco Tomassini, Sébastien Verel

Fitness Clouds and Problem Hardness in Genetic Programming

Leonardo Vanneschi[2], Manuel Clergue[1], Philippe Collard[1], Marco Tomassini[2], and Sébastien Vérel[1]

[1] I3S Laboratory, University of Nice, Sophia Antipolis, France
[2] Information Systems Department, University of Lausanne, Lausanne, Switzerland

Abstract. This paper presents an investigation of genetic programming fitness landscapes. We propose a new indicator of problem hardness for tree-based genetic programming, called *negative slope coefficient*, based on the concept of fitness cloud. The negative slope coefficient is a predictive measure, i.e. it can be calculated without prior knowledge of the global optima. The fitness cloud is generated via a sampling of individuals obtained with the Metropolis-Hastings method. The reliability of the negative slope coefficient is tested on a set of well known and representative genetic programming benchmarks, comprising the binomial-3 problem, the even parity problem and the artificial ant on the Santa Fe trail.

1 Introduction

Genetic Programming (GP) has had an indeniable practical success in its fifteen years of existence [13,14]. However, it is still difficult to understand why some problems are easily solved by GP, while others resist solution or require massive amounts of computational effort. It would thus be of interest if we were able to somehow classify problems according to some measure of difficulty. To start with, it might be useful to take a look at what has been done in the older field of *genetic algorithms* (GAs). Difficulty studies in GAs have been pioneered by Goldberg and coworkers [4,6,8]. Their approach is focused on the construction of functions that should *a priori* be easy or hard for GAs to solve. These ideas have been followed by many others, for instance [18,5] and have been at least partly successful in the sense that they have been the source of many ideas as to what makes a problem easy or difficult for GAs. One concept that underlies many of these approaches is based on the notion of *fitness landscape*. The metaphor of a fitness landscape [22], although not without faults, has been a fruitful one in several fields. In particular, a statistic called *fitness distance correlation* (FDC) [10] has been studied in detail in the past in the context of GAs. Its suitability for GP has been investigated in [2,24,23]. As far as GP is concerned, but also in the GA field, the general conclusion of these studies was that the FDC can be considered as a rather reliable indicator of problem hardness. However, it has some severe drawbacks: sometimes the measure does not give any indication, and problems can be constructed for which the FDC leads to contradictory conclusions. The first consideration is not really serious since it manifests itself rarely and other tools, such as the analysis of the fitness-distance scatterplot can be brought to bear in these cases. On the other hand, the existence of counterexamples casts a shadow on the

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 690–701, 2004.

usefulness of the FDC, although such cases are typically contrived ones and they do not seem to appear often among "natural" problems. But the really annoying fact about FDC, and its main weakness in our opinion, is that the optimal solution (or solutions) must be known beforehand, which is obviously unrealistic in applied search and optimization problems, and prevents us from applying FDC to many GP benchmarks and real-life applications. Thus, although the study of FDC is an useful first step, we present another approach based on quantities that can be measured without any explicit knowledge of the genotype of optimal solutions, such as different kinds of fitness distributions. A second consideration concerns the way in which space is sampled: uniform random sampling has the merit of being simple and algorithm-independent (only random search is implied), but it would be more useful to have a sample of the landscape as "seen" by a specific algorithm for it is the latter the one that is really relevant. In this way, more weight can be given to particular points in the space. Thus, sampling according to a given stationary probability distribution, like Markov chain Monte Carlo, appears to be more appropriate.

This paper is structured as follows: section 2 summarizes some techniques used by other researchers in the past few years, which inspired this work. Section 3 introduces the concept of fitness cloud. This concept is used in section 4 to define a new measure, called *negative slope coefficient* (NSC), that is proposed as an indicator of problem hardness. Section 5 briefly introduces the test problems used to verify the reliability of NSC. Section 6 shows experimental results. Finally, section 7 offers our conclusions and hints for future work.

2 Fitness-Fitness Correlation: Previous Work

In genetic algorithms, plotting fitness against some features is not a new idea. Manderick *et al*. [17] study the correlation coefficient of genetic operators: they compute the correlation between the fitnesses of a number of parents and the fitnesses of their offspring. Grefenstette [7] uses fitness distribution of genetic operators to predict GA behaviour. Rosé *et al.* [20] develop the *density of states* approach by plotting the number of genotypes with the same fitness value. Smith *et al.* [21] focus on notions of *evolvability* and *neutrality*; they plot the average fitness of offspring over fitness according to Hamming neighbouring. *Evolvability* refers to the efficiency of evolutionary search. It is defined by Altenberg as "the ability of an operator/representation scheme to produce offspring that are fitter than their parents" [1].

Fitness correlation measures are almost absent in GP but there are a few precursors. The first work we are aware of is the article of Kinnear [12] in which GP difficulty is analysed through the use of the fitness *autocorrelation function*, as first proposed by Weinberger [26]. While fitness autocorrelation analysis has been useful in the study of NK landscapes [26], Kinnear found his results inconclusive and difficult to interpret: essentially no simple relationship was found between correlation length values and GP hardness. The work of Nikolaev and Slavov [19] also makes use of the fitness autocorrelation function with the main purpose of determining which mutation operator, among a few that they propose, "sees" a smoother landscape on a particular problem. Their analysis however, does not lead to a general study of problem difficulty in GP, which is our aim here. More recently, a much more detailed study of fitness correlation in GP has

been presented by Igel and Chellapilla [9]. In this work, several fitness-fitness probability distributions are analyzed on four typical GP problems as tools for understanding the effect of variation operators on the search. They favor using a combination of variation operators rather than a single one, and suggest that fitness distributions measures might be used at execution time to dynamically set the operator probabilities in order to be more exploitative or explorative. An off-line analysis of the results is also hinted at, leading to a possibly better operator choice for similar problems. Although interesting, this work does not really deal with GP difficulty; it is rather an attempt to automatically tune the evolutionary search and determine the most efficient variation operators for a given problem class. Some GP lansdcapes have been systematically investigated by Langdon and Poli [15]. Their results will be compared with our findings in some cases (see section 6).

3 Fitness Cloud

In this section we use the fitness cloud metaphor, first introduced in [25] by Vérel and coworkers for binary landscapes. In order to get a visual rendering of evolvability, for each string x in the genotype space a point is plotted, the abscissa of which is its fitness value f, and the ordinate the fitness $\tilde{f}$ of a particular neighbour that can be chosen in many different ways. The result is a scatterplot, that was called the *fitness cloud* in [25].

3.1 Evolvability

One feature that is intuitively linked, although not exactly identical, to problem difficulty is evolvability, i.e. the capacity of genetic operators to improve fitness quality. The most natural way to study evolvability is to plot the fitness values of individuals against the fitness values of their neighbors, where a neighbor is obtained by applying one step of a genetic operator to the individual. The genetic operator used here is standard subtree mutation [13], i.e. a random subtree of a selected individual is replaced by a randomly generated tree.

Formally, let Γ be the whole search space of a GP problem and $V(\gamma)$ the set of all the neighbors of individual $\gamma \in \Gamma$, obtained by applying one step of standard subtree mutation to it. The subtree mutation operator is not allowed to choose the root node as its insertion point, thus $V(\gamma)$ is different from the entire search space.

Now let f be the fitness function of the problem at hand. We define the following set of points on a bidimensional plane: $S = \{(f(\gamma), f(\nu)), \forall \gamma \in \Gamma, \forall \nu \in V(\gamma)\}$. The graphical representation of S is the scatterplot of the fitness of all the individuals belonging to the search space vs. the fitness of all their neighbors. We hypothesize that the shape of this scatterplot can give an indication of the evolvability of the genetic operators used and thus some hints about the difficulty of the problem at hand.

The fitness cloud implicitly gives some insight on the genotype to phenotype map. The set of genotypes that all have equal fitness is a *neutral set* [11]. Such a set corresponds to one abscissa in the fitness/fitness plan; according to this abscissa, a vertical slice from the cloud represents the set of fitnesses that could be reached from this set of neutrality. For a given offspring fitness value $\tilde{f}$, an horizontal slice represents all the fitness values from which one can reach $\tilde{f}$.

3.2 Sampling Methodology

In general, the size of the search space doesn't allow to consider all the possible individuals. Thus, we need to use samples. Sampling the program space according to a uniform probability distribution gives the same weight to all the sampled points. However, as it happens, many of those points are not really significant from the problem space point of view. For example, we might be repeatedly sampling points belonging to the same plateau of fitness, which may be wasted effort. For this reason, we prefer to sample the space according to a distribution that gives more weight to "important" values in the space, for instance those at a higher fitness level. This is also the case of any biased searcher such as an evolutionary algorithm, simulated annealing and other heuristics, and thus the sampling process more closely simulates the way in which the program space would be traversed by a searcher. This is a standard problem and it is well known that the sampling can be done by using Metropolis method [16] or any other equivalent importance sampling technique employed in simulation. Here we use *Metropolis-Hastings* sampling, that is an extension of Metropolis to non-symmetric stationary probability distributions. This technique can be defined as follows. Let α be the function defined as:

$$\alpha(x, y) = min\{1, \frac{y}{x}\},$$

and $f(\gamma_k)$ be the fitness of individual γ_k. A sample of GP individuals $\{\gamma_1, \gamma_2, \ldots, \gamma_n\}$ is built with the following algorithm:

begin
  γ_1 `is generated uniformly at random;`
  for $k := 2$ **to** n **do**
    `1. an individual` δ `is generated uniformly at random;`
    `2. a random number` u `is generated from a`
      `uniform` $(0, 1)$ `distribution;`
    `3.` **if** $(u \leq \alpha(f(\gamma_{k-1}), f(\delta)))$
      then $\gamma_k := \delta$
      else goto `1.`
      endif
    `4.` $k := k + 1$`;`
  endfor
end

For each sampled point, the neighbors of that point must be generated. The number of all the possible neighbors of a tree depends on the particular genetic operator used. For standard subtree mutation, as used here, this number is huge. On the other hand, studying all possible neighbors of each sampled individual is worthless since most of them will be discarded by selection as soon as they are created. Thus, instead of studying the set $V(\gamma)$ of all the neighbors of each sampled individual γ, we only consider a subset of size $q \ll |V(\gamma)|$, obtained by applying tournament selection to its elements: to obtain each one of these q neighbors, a set of r random neighbors is generated (where r is the tournament size), by applying one step of subtree mutation to γ, and the best one is then chosen.

The terminology of section 3.1 is thus updated as follows: we refer to Γ as a sample of individuals obtained with the Metropolis-Hastings technique and, for each γ belonging to Γ, we refer to $V(\gamma)$ as a subset of size q of its neighbors, obtained by the application of the tournament selection mechanism.

4 Negative Slope Coefficient

The fitness cloud can be of help in determining some characteristics of the fitness landscape related to evolvability and problem difficulty. But the mere observation of the scatterplot is not sufficient to quantify these features. In this section, we present an algebraic measure, called *negative slope coefficient* (NSC), that we propose as a new indicator of problem difficulty for GP.

The abscissas of a scatterplot can be partitioned into m segments $\{I_1, I_2, \ldots, I_m\}$ of the same length. Analogously, a partition of the ordinates $\{J_1, J_2, \ldots, J_m\}$ can be done, where each segment J_i contains all the ordinates corresponding to the abscissas contained in I_i.

Let $M_1, M_2, \ldots, M_m$ be the averages of the abscissa values contained inside the segments $I_1, I_2, \ldots, I_m$ and let $N_1, N_2, \ldots, N_m$ be the averages of the ordinate values in $J_1, J_2, \ldots, J_m$. Then we can define the set of segments $\{S_1, S_2, \ldots, S_{m-1}\}$, where each S_i connects the point (M_i, N_i) to the point (M_{i+1}, N_{i+1}). For each one of these segments S_i, the *slope* P_i can be calculated as follows:

$$P_i = \frac{N_{i+1} - N_i}{M_{i+1} - M_i}$$

Finally, we can define the negative slope coefficient as:

$$\text{NSC} = \sum_{i=1}^{m-1} min\ (P_i,\ 0)$$

We hypothesize that NSC is an indicator of problem difficulty in the following sense: if NSC$= 0$, the problem is easy, if NSC< 0 the problem is difficult and the magnitude of NSC quantifies this difficulty: the more negative its value, the more difficult the problem. In other words, according to our hypothesis, a problem is difficult if at least one of the segments $S_1, S_2, \ldots, S_{m-1}$ has a negative slope and the sum of all the negative slopes gives a measure of problem hardness. The idea is that the presence of a segment with negative slope indicates a bad evolvability for individuals having fitness values contained in that segment. Note that, for the time being, we didn't try to normalize NSC values to a given range. This means that NSC results for different problems are not comparable among them.

In the following sections, NSC is tested as a measure of problem hardness on a set of well known GP benchmarks. To be able to validate difficulty predictions, we define a *performance* measure, as being the proportion of GP runs for which the global optimum has been found in less than 500 generations over 100 independent executions. Good or bad performance values correspond to our intuition of what "easy" or "hard" means in practice. All GP runs executed to calculate performance values have used the same

set of GP parameters: generational GP, population size of 200 individuals, standard GP mutation used as the sole genetic operator with a rate of 95%, tournament selection of size 10, ramped half and half initialization, maximum depth of individuals specified in the following case by case, elitism (i.e. survival of the best individual into the newly generated population).

For the sake of comparison we also measure the *fitness-fitness correlation* (FFC) [17]. Given the set $X = \{x_1, x_2, ..., x_n\}$ of all the abscissas of a scatterplot and the set $Y = \{y_1, y_2, ..., y_n\}$ of all the ordinates of the same scatterplot, the FFC is defined as: $C_{XY}/\sigma_X\sigma_Y$, where $C_{XY} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})$ is the covariance of X and Y, and σ_X, σ_Y, $\overline{x}$, $\overline{y}$ are the standard deviations and means of X and Y.

5 Test Problems

Problems used in this work are briefly described below. For a more detailed description, see [3,13]. Except for the ant problem, which is included because of the large body of knowledge accumulated on it, the other problems have been chosen because they are representative of important problem classes (respectively symbolic regression and boolean), and their difficulty can be tuned.

The binomial-3 problem. This benchmark (first introduced by Daida *et al.* in [3]) is an instance of the well known symbolic regression problem. The function to be approximated is $f(x) = 1 + 3x + 3x^2 + x^3$. Fitness cases are 50 equidistant points over the range $[-1, 0)$. Fitness is the sum of absolute errors over all fitness cases. A hit is defined as being within 0.01 in ordinate for each one of the 50 fitness cases. The function set is $\mathcal{F} = \{+, -, *, //\}$, where $//$ is the protected division, i.e. it returns 1 if the denominator is 0. The terminal set is $\mathcal{T} = \{x, \mathcal{R}\}$, where x is the symbolic variable and $\mathcal{R}$ is the set of ephemeral random constants (ERCs). ERCs are uniformly distributed over a specified interval of the form $[-a_{\mathcal{R}}, a_{\mathcal{R}}]$, they are generated once at population initialization and they are not changed in value during the course of a GP run. According to Daida and coworkers [3], difficulty tuning is achieved by varying the value of $a_{\mathcal{R}}$.

The even parity k problem. The boolean even parity k function [13] of k boolean arguments returns *true* if an even number of its boolean arguments evaluates to true, otherwise it returns *false*. The number of fitness cases to be checked is 2^k. Fitness is computed as 2^k minus the number of hits over the 2^k cases. Thus a perfect individual has fitness 0, while the worst individual has fitness 2^k. The set of functions we employed is $\mathcal{F} = \{NAND, NOR\}$. The terminal set is composed of k different boolean variables. Difficulty tuning is achieved by varying the value of k.

Artificial ant on the Santa Fe trail. In this problem, an artificial ant is placed on a 32×32 toroidal grid. Some of the cells from the grid contain food pellets. The goal is to find a navigation strategy for the ant that maximizes its food intake. We use the same set of functions and terminals as in [13] and the same trail definition. As fitness function, we use the total number of food pellets lying on the trail (89) minus the amount of food eaten by the ant during his path. This turns the problem into a minimization one, like the previous ones.

6 Experimental Results

6.1 The Binomial-3 Problem

Figure 1 shows the scatterplots and the set of segments $\{S_1, S_2, \ldots, S_m\}$ as defined in section 4 (with $m = 10$) for the binomial-3 problem with $a_{\mathcal{R}} = 1$ (figure 1(a)), $a_{\mathcal{R}} = 10$ (figure 1(b)), $a_{\mathcal{R}} = 100$ (figure 1(c)) and $a_{\mathcal{R}} = 1000$ (figure 1(d)). Parameters used are as follows: maximum tree depth $= 26$, $|\Gamma| = 40000$, i.e. a sample of 40000 individuals has been used, obtained with the Metropolis-Hastings sampling, $\forall \gamma \in \Gamma \;\; |V(\gamma)| = 1$, i.e. for each sampled individual, only one neighbor has been considered. It has been obtained by one step of tournament selection and standard subtree mutation has been used as the operator to generate the neighborhood.

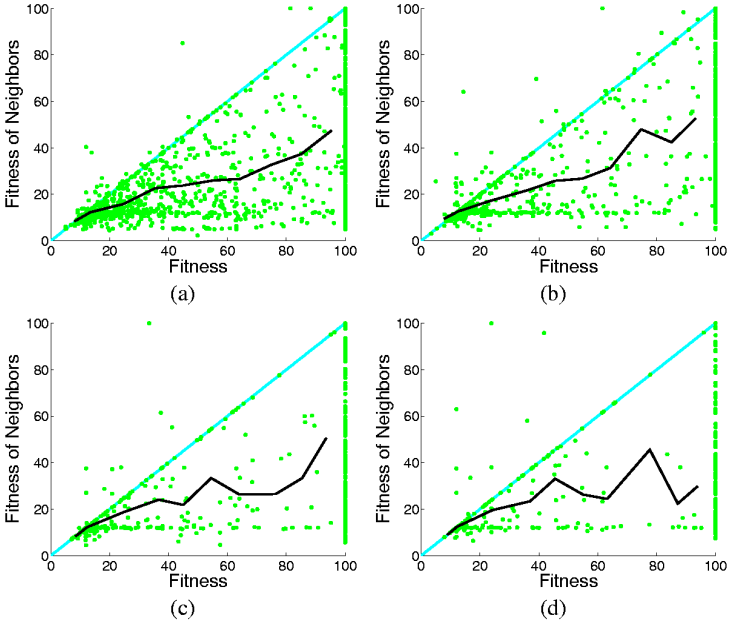

Fig. 1. Binomial-3 results. (a): $a_{\mathcal{R}} = 1$. (b): $a_{\mathcal{R}} = 10$. (c): $a_{\mathcal{R}} = 100$. (d): $a_{\mathcal{R}} = 1000$. Fitness clouds have been obtained by a sample of 40000 individuals.

Table 1 shows some data about these experiments. Column one of table 1 represents the corresponding scatterplot in figure 1. Column two contains the $a_{\mathcal{R}}$ value. Column three contains performance. Columns four and five contain values of NSC and FFC respectively.

Table 1. Binomial-3 problem. Some data related to scatterplots of figure 1.

scatterplot	$a_{\mathcal{R}}$	p	NSC	FFC
fig. 1(a)	1	0.89	0	0.70
fig. 1(b)	10	0.42	-0.53	0.74
fig. 1(c)	100	0.35	-1.01	0.75
fig. 1(d)	1000	0.29	-3.39	0.75

Table 2. Even parity. Indicators related to scatterplots of figure 2.

scatterplot	problem	p	NSC	FFC
fig. 2(a)	even parity 3	0.98	0	0.56
fig. 2(b)	even parity 5	0.01	-0.11	0.39
fig. 2(c)	even parity 7	0	-0.49	0.25
fig. 2(d)	even parity 9	0	-0.55	0.23

These results show that NSC values get smaller as the problem becomes harder, and it is zero when the problem is easy ($a_{\mathcal{R}} = 1$). On the other hand, FFC doesn't seem to give any indication about problem difficulty, which confirms Kinnear's observations [12]. Moreover, the points in the scatterplots seem to cluster around good (i.e. small) fitness values as the problem gets easier (remark: points having a fitness value above 100 have not been visualized in the scatterplots of figure 1 for the sake of clarity, even though, of course, they have been used to calculate the NSC). In conclusion, the presence or absence of segments with negative slope (quantified by the NSC), in conjunction with the graphical representation of the scatterplot, seems to be useful to estimate problem hardness.

6.2 The Even Parity k Problem

Figure 2 shows the scatterplots and the set of segments $\{S_1, S_2, \ldots, S_m\}$ (where m has been set to 6) for the even parity 3, even parity 5, even parity 7 and even parity 9 problems. Parameters used are as follows: maximum tree depth $= 10$, $|\Gamma| = 40000$ obtained with the Metropolis-Hastings sampling, $\forall \gamma \in \Gamma \;\; |V(\gamma)| = 1$. Tournament has been used as a selection mechanism and standard subtree mutation as the operator for generating the neighborhood.

Table 2 shows some data about these experiments with the same notation and meaning as in table 1, except that column two now refers to the problem rank. Analogously to what happens for the binomial-3 problem, NSC values get smaller as the problem becomes harder, they are always negative for hard problems, and zero for easy ones. Once again the points in the scatterplots seem to cluster around good fitness values as the problem gets easier. These results are in qualitative agreement with intuition and with those found by other researchers (e.g. [13]). Thus, the utility of NSC as an indicator of problem hardness, together with the graphic representation of the fitness cloud, is confirmed.

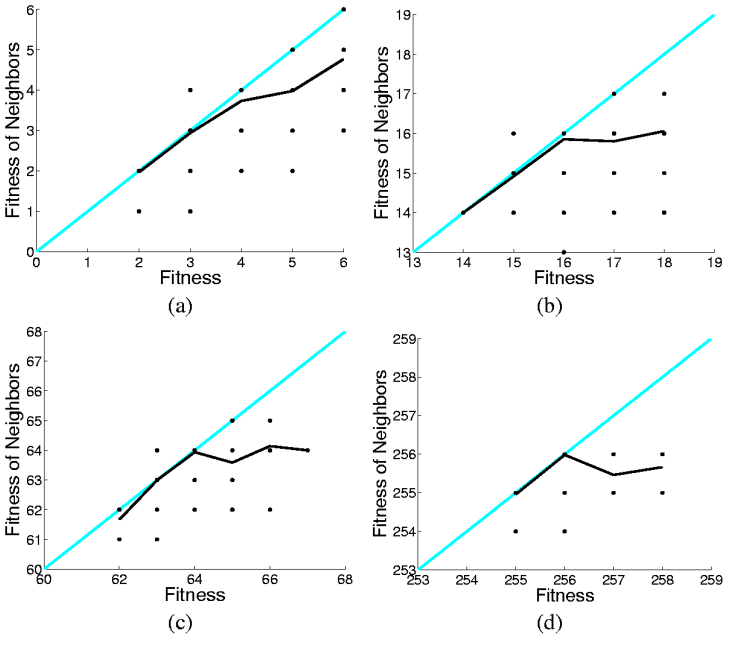

Fig. 2. Results for the even parity k problem. (a): Even parity 3. (b): Even parity 5. (c): Even parity 7. (d): Even parity 9. Fitness clouds have been obtained by a sample of 40000 individuals. Note that many sampled individuals may have the same fitness value.

6.3 The Artificial Ant on the Santa Fe Trail

This problem, among others, has been studied in depth by Langdon and Poli. In [15] they did a detailed analysis of the problem's fitness landscape by enumeration for small programs, and by sampling for bigger program sizes. They also studied the problem from the point of view of the schema theory and found it to be deceptive. Figure 3 shows the scatterplots and the set of segments $\{S_1, S_2, \ldots, S_m\}$ (where m has been set to 10) for the artificial ant problem with maximum tree depths equal to 10 and 6 respectively. The other parameters used are: $|\Gamma| = 40000$ obtained with the Metropolis-Hastings sampling, $\forall \gamma \in \Gamma \;\; |V(\gamma)| = 1$. Tournament has been used as a selection mechanism and standard subtree mutation as the operator to generate the neighborhood.

Table 3 shows the results of the experiments, where column two indicates the maximum program depth allowed, and the other values have the usual meaning. Both problems turn out to be difficult and NSC is negative for both problem instances. Indeed, there are many local optima in the ant space and the search can become easily trapped in one of them. Larger programs appear to be slightly easier to search for a solution than smaller ones. This is in agreement with Langdon's and Poli's findings which say that the number of solutions increases exponentially with program size. The unusual shape of the scatterplot and the fact that some mean segments are horizontal can also be accounted for in terms of Langdon's and Poli's analysis. They found that, for a given program, most

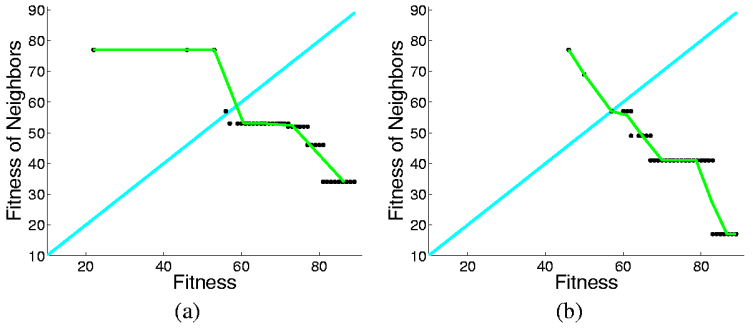

Fig. 3. (a): Artificial Ant problem with maximum tree depth equal to 10. (b): Artificial Ant problem with maximum tree depth equal to 6. Fitness clouds have been obtained by a sample of 40000 individuals. Note that many sampled individuals may have the same fitness value.

neighbors have the same fitness or are worse; a fact that is confirmed here. Moreover, the phenomenon is more marked for longer programs, again confirmed by figure 3. We also see that the fitter the individual, the more difficult becomes to improve it, a fact that is represented by the large negative slope segment at low fitness (remember that we have defined fitness so that low values are better). This too is in agreement with the analysis in [15].

Table 3. Ant problem results. Some data related to scatterplots of figure 3.

scatterplot	max. tree depth	p	NSC	FFC
fig. 3(a)	10	0.05	-6.06	-0.88
fig. 3(b)	6	0	-11.42	-0.82

7 Conclusions and Future Work

A new indicator of problem hardness for GP, called negative slope coefficient, is proposed in this work. This measure is based on the concept of fitness cloud, that visualizes on a plane the relationship between the fitness values of a sample of individuals and the fitness of some of their neighbors. Sampling has been done with the Metropolis-Hastings technique, so as to give more weight to individuals with good fitness and to preserve the original distribution of the fitness function over all the search space. Contrary to fitness distance correlation, NSC is predictive, i.e. it can be used without prior knowledge of the global optima. Thus, NSC can be used to quantify difficulty of standard GP benchmarks, where the cardinality and shape of global optima is not known *a priori*. Studies over three well-known GP benchmarks (binomial-3, even parity and artificial ant) have been presented here, confirming the suitability of NSC as measure of problem

hardness, at least for the cases studied. Such a systematic study of GP problem hardness, performed using a well defined algebraic measure, had, to our knowledge, never been applied to these benchmarks before. Other experiments on these benchmarks using a different sampling technique (standard Metropolis) and on other classes of tunably difficult problems (like trap functions and royal trees) have been done. Fitness clouds using more than one neighbor for each sampled individual have been analysed. Moreover, fitness clouds obtained by applying more than once the mutation operator at each sampled individual have been studied. Finally, a less disruptive mutation operator (structural mutation introduced in [24]) has been used. All these results (not shown here for lack of space) confirm the suitability of NSC as an indicator of problem hardness.

Future work includes a more exhaustive study of NSC and other measures based on fitness clouds over a wider set of GP benchmarks, the use of more sophisticated sampling techniques and the study of techniques to automatically define some quantities that in this paper have been chosen somehow arbitrarily, such as the number of segments in which the fitness cloud is partitioned and their size. Furthermore, we will study NSC from a statistical point of view, on order to understand the significance of the results obtained and how they change from one sampling to the other. Since we don't believe NSC to be infallible, as is true for any statistic based on samples, these studies should also lead to the discovery of some drawbacks of this measure that should inspire future extensions.

Acknowledgments. L. Vanneschi and M. Tomassini gratefully acknowledge financial support by the Fonds National Suisse pour la recherche scientifique under contract 200021-100112/1.

References

1. L. Altenberg. The evolution of evolvability in genetic programming. In K. Kinnear, editor, *Advances in Genetic Programming*, pages 47–74, Cambridge, MA, 1994. The MIT Press.
2. M. Clergue, P. Collard, M. Tomassini, and L. Vanneschi. Fitness distance correlation and problem difficulty for genetic programming. In W. B. Langdon et. al., editor, *Proceedings of the genetic and evolutionary computation conference GECCO'02*, pages 724–732. Morgan Kaufmann, San Francisco, CA, 2002.
3. J. M. Daida, R. Bertram, S. Stanhope, J. Khoo, S. Chaudhary, and O. Chaudhary. What makes a problem GP-hard? analysis of a tunably difficult problem in genetic programming. *Genetic Programming and Evolvable Machines*, 2:165–191, 2001.
4. K. Deb and D. E. Goldberg. Analyzing deception in trap functions. In D. Whitley, editor, *Foundations of Genetic Algorithms, 2*, pages 93–108. Morgan Kaufmann, 1993.
5. S. Forrest and M. Mitchell. What makes a problem hard for a genetic algorithm? some anomalous results and their explanation. *Machine Learning*, 13:285–319, 1993.
6. D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Boston, MA, 1989.
7. J. Grefenstette. Predictive models using fitness distributions of genetic operators. In D. Whitley and M. Vose, editors, *Foundations of Genetic Algorithms, 3*, pages 139–161. Morgan Kaufmann, 1995.

8. J. Horn and D. E. Goldberg. Genetic algorithm difficulty and the modality of the fitness landscapes. In D. Whitley and M. Vose, editors, *Foundations of Genetic Algorithms, 3*, pages 243–269. Morgan Kaufmann, 1995.
9. C. Igel and K. Chellapilla. Fitness distributions: Tools for designing efficient evolutionary computations. In L. Spector, W. B. Langdon, U.-M. O'Reilly, and P. Angeline, editors, *Advances in Genetic Programming 3*, pages 191–216, Cambridge, MA, 1999. The MIT Press.
10. T. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search*. PhD thesis, University of New Mexico, Albuquerque, 1995.
11. M. Kimura. *The Neutral Theory of Molecular Evolution*. Cambridge University Press, Cambridge, UK, 1983.
12. K. E. Kinnear. Fitness landscapes and difficulty in genetic programming. In *Proceedings of the First IEEEConference on Evolutionary Computing*, pages 142–147. IEEE Press, Piscataway, NY, 1994.
13. J. R. Koza. *Genetic Programming*. The MIT Press, Cambridge, Massachusetts, 1992.
14. J. R. Koza, M. J. Streeter, and M. A. Keane. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*. Kluwer Academic Publishers, Boston, MA, 2003.
15. W. B. Langdon and R. Poli. *Foundations of Genetic Programming*. Springer, Berlin, 2002.
16. N. Madras. *Lectures on Monte Carlo Methods*. American Mathematical Society, Providence, Rhode Island, 2002.
17. B. Manderick, M. de Weger, and P. Spiessens. The genetic algorithm and the structure of the fitness landscape. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 143–150. Morgan Kaufmann, 1991.
18. M. Mitchell, S. Forrest, and J. Holland. The royal road for genetic algorithms: fitness landscapes and ga performance. In F. J. Varela and P. Bourgine, editors, *Toward a Practice of Autonomous Systems, Proceedings of the First European Conference on Artificial Life*, pages 245–254. The MIT Press, 1992.
19. N. I. Nikolaev and V. Slavov. Concepts of inductive genetic programming. In W. Banzhaf et. al., editor, *Genetic Programming, Proceedings of EuroGP'1998*, volume 1391 of *LNCS*, pages 49–59. Springer-Verlag, 1998.
20. H. Rosé, W. Ebeling, and T. Asselmeyer. The density of states - a measure of the difficulty of optimisation problems. In H.-M. Voigt et al., editor, *Parallel Problem Solving from Nature - PPSN IV*, volume 1141 of *Lecture Notes in Computer Science*, pages 208–217. Springer-Verlag, Heidelberg, 1996.
21. Smith, Husbands, Layzell, and O'Shea. Fitness landscapes and evolvability. *Evolutionary Computation*, 1(10):1–34, 2001.
22. P. F. Stadler. Fitness landscapes. In M. Lässig and Valleriani, editors, *Biological Evolution and Statistical Physics*, volume 585 of *Lecture Notes Physics*, pages 187–207, Heidelberg, 2002. Springer-Verlag.
23. L. Vanneschi, M. Tomassini, M. Clergue, and P. Collard. Difficulty of unimodal and multimodal landscapes in genetic programming. In E. Cantú-Paz et. al., editor, *Genetic and Evolutionary Computation – GECCO-2003*, volume 2724 of *LNCS*, pages 1788–1799. Springer-Verlag, Berlin, 2003.
24. L. Vanneschi, M. Tomassini, P. Collard, and M. Clergue. Fitness distance correlation in structural mutation genetic programming. In C. Ryan et al., editor, *Genetic Programming, 6th European Conference, EuroGP2003*, volume 455–464 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg, 2003.
25. S. Vérel, P. Collard, and M. Clergue. Where are bottleneck in nk-fitness landscapes ? In *CEC 2003: IEEE International Congress on Evolutionary Computation. Canberra, Australia*, pages 273–280. IEEE Press, Piscataway, NJ, 2003.
26. E. D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biol. Cybern.*, 63:325–336, 1990.